

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-006614

(43)Date of publication of application : 10.01.1997

(51)Int.Cl.

G06F 9/38

G06F 9/30

(21)Application number : 07-154695

(71)Applicant : SANYO ELECTRIC CO LTD

(22)Date of filing : 21.06.1995

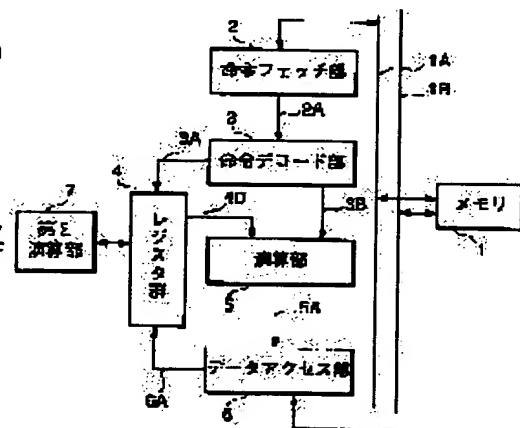
(72)Inventor : MIURA HIROKI
KOMURA YASUTO
MATSUMOTO KENJI

(54) DATA PROCESSOR

(57)Abstract:

PURPOSE: To improve performance and to reduce a program size.

CONSTITUTION: An instruction fetch part 2 reads in an instruction and data from a memory 1, and an instruction decoding part 3 decodes them. The operation of a second operation part 7 is started by a data transfer instruction to an input register of a register group 4, and the operation result is stored in an output register. When unprocessed data remains in the input or output register, the succeeding data processing is stopped. Plural sets of input and output registers and computing elements are provided and are selected in accordance with the processing. Since prescribed operation is executed by the data transfer instruction, the processing speed is increased and the program is simplified. The extensibility is high because the addition of a special operation instruction is unnecessary.



LEGAL STATUS

[Date of request for examination]

31.05.2002

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-6614

(43) 公開日 平成9年(1997)1月10日

(51) Int.Cl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 9/38	3 7 0		G 0 6 F 9/38	3 7 0 C
9/30	3 7 0		9/30	3 7 0

審査請求 未請求 請求項の数11 O L (全 13 頁)

(21) 出願番号 特願平7-154695

(22) 出願日 平成7年(1995)6月21日

(71) 出願人 000001889

三洋電機株式会社

大阪府守口市京阪本通2丁目5番5号

(72) 発明者 三浦 宏喜

大阪府守口市京阪本通2丁目5番5号 三洋電機株式会社内

(72) 発明者 甲村 康人

大阪府守口市京阪本通2丁目5番5号 三洋電機株式会社内

(72) 発明者 松本 健志

大阪府守口市京阪本通2丁目5番5号 三洋電機株式会社内

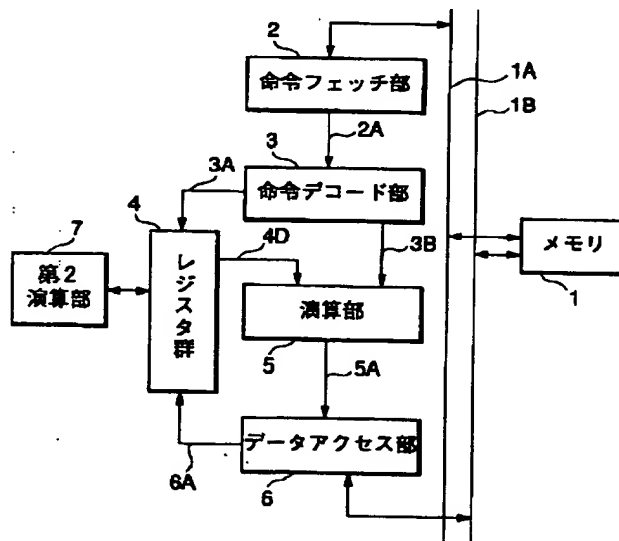
(74) 代理人 弁理士 吉田 研二 (外2名)

(54) 【発明の名称】 データ処理装置

(57) 【要約】

【目的】 性能向上とプログラムサイズの縮小。

【構成】 命令フェッチ部2がメモリ1から命令とデータを読み込み、命令デコード部3で解読する。レジスタ群4の入力用レジスタへのデータ転送命令によって第2演算部7の演算が起動され、演算結果は出力用レジスタに格納される。入力用または出力用レジスタに未処理のデータが残っていれば、後続のデータ処理を停止する。入力用、出力用レジスタセット、演算器を複数設け、処理に応じて択する。データ転送命令で所期の演算が実行されるため、処理の高速化、プログラムの簡素化が可能となる。特別な演算命令の追加が不要のため、拡張性も高い。



【特許請求の範囲】

【請求項1】 各種命令を実行してデータ処理を行うデータ処理装置であって、

演算命令の実行によって所定の演算処理を起動する第1演算手段と、

所定の入力用レジスタ及び出力用レジスタを含み、データ転送命令によるアクセスが可能な複数のレジスタと、データ転送命令の実行によって前記入力用レジスタに対するデータ転送が行われたことを契機としてそのデータに対する所定の演算処理を起動し、その演算結果データを前記出力用レジスタに書き込む第2演算手段と、前記演算結果データの読み出しが行われるまで、前記出力用レジスタに対する後続の演算結果データの書き込みを待たせる書込停止手段と、を備えたことを特徴とするデータ処理装置。

【請求項2】 請求項1に記載のデータ処理装置において、該装置はさらに、

命令の実行が前記演算結果データの読み出しを伴うとき、所望の演算結果データが前記出力用レジスタに書き込まれるまで読み出しを待たせる読出停止手段を含むことを特徴とするデータ処理装置。

【請求項3】 請求項1に記載のデータ処理装置において、該装置はさらに、

前記書込停止手段によって書き込みが待たされているときに、前記入力用レジスタに対するデータ転送が行われた場合、前記第2演算手段による演算処理の起動を待たせる演算停止手段を含むことを特徴とするデータ処理装置。

【請求項4】 請求項1に記載のデータ処理装置において、該装置はさらに、

前記入力用レジスタに有効なデータが保持されているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送を待たせる転送停止手段を含むことを特徴とするデータ処理装置。

【請求項5】 請求項1に記載のデータ処理装置において、

該装置は、前記入力用レジスタ及び出力用レジスタの組合せによる複数のレジスタセットを備え、さらにこれら複数のレジスタセットから演算処理を行うべきレジスタセットを指定する指定手段を有することを特徴とするデータ処理装置。

【請求項6】 請求項1に記載のデータ処理装置において、

前記第2演算手段は、それぞれが異なる演算を行う複数の演算器を有し、

該装置は、これら複数の演算器のうちデータ転送が行われたとき実際に演算処理を起動すべきものを指定する指定手段を有することを特徴とするデータ処理装置。

【請求項7】 請求項6に記載のデータ処理装置において、

前記指定手段は、前記入力用レジスタおよび出力用レジスタとは別に設けられた制御レジスタであることを特徴とするデータ処理装置。

【請求項8】 請求項1に記載のデータ処理装置において、

前記第2演算手段は、それぞれが異なる演算を行う複数の演算器を有し、

該装置は、

前記複数の演算器のうち1つの演算器の演算結果データを別の演算器へ入力する手段と、

前記出力用レジスタに書き込まれたデータを前記別の演算器に入力する手段と、

前記別の演算器の演算結果データを前記出力用レジスタに再帰的に入力する手段と、

を有することを特徴とするデータ処理装置。

【請求項9】 請求項1に記載のデータ処理装置において、

該装置は、前記入力用レジスタを複数備え、

これらの入力用レジスタに対して順次データ転送がなされるとき、1回の演算処理に必要な複数のデータが入力用レジスタに揃うタイミングを検出する検出手段と、

これらのデータが揃うまで、前記第2演算手段による演算処理の起動を待たせる演算停止手段を含むことを特徴とするデータ処理装置。

【請求項10】 請求項8に記載のデータ処理装置において、

前記入力用レジスタと前記第2演算手段との間に介挿され、前記入力用レジスタに転送されたデータを順次記憶するFIFOメモリと、

前記FIFOメモリからのデータ読み出しを制御する制御手段と、

を有することを特徴とするデータ処理装置。

【請求項11】 請求項9に記載のデータ処理装置において、

前記制御手段は、演算処理に必要な複数のデータが前記FIFOに揃ったとき前記FIFOメモリからのデータ読み出しを行い、

前記演算停止手段は、前記FIFOメモリからデータが読み出されたとき前記第2演算手段による演算処理の起動を許可することを特徴とするデータ処理装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、各種命令を実行してデータ処理を行うデータ処理装置に関する。

【0002】

【従来の技術】マイクロプロセッサを代表とするデータ処理装置として、メモリに格納された命令を順次フェッチ、デコードして実行する制御駆動型の装置が数多く存在する。

【0003】図1はRISC (reduced instruction se

t computer) 方式のプロセッサの構成図である。このプロセッサは、メモリ1、命令フェッチ部2、命令デコード部3、レジスタ群4、演算部5、データアクセス部6を持つ。この構成において、メモリ1から順次フェッチされる演算命令、データ転送(ロード/ストア)命令などがデコードされ、演算部5において、命令コードで指示される演算が実施される。

【0004】従来、図1の演算部5の中に、加算、減算、論理和、論理積などの演算を実行する基本的な算術論理演算部(ALU)と別に、例えば乗算を高速に実行するための乗算器を並列的に備えるものが存在した。また、整数演算用のALUと浮動小数点演算用のALUが並設されたものもあった。演算部5を構成する各種演算器を利用してデータ処理を行う場合、演算器による演算処理に対応する演算命令を設け、これをプログラム中に記述してプロセッサに読み込む方法が一般的だった。すなわち、例えばADD(整数加算)、SUB(整数減

```
LD      R2, (R0)
LD      R3, (R1)
MUL     R2, R3
```

この結果、以下の課題があった。

【0007】[課題1] 処理性能の限界

仮に、1命令サイクルで演算を実行する高速の演算器を搭載しても、全体の処理レートとしては、3命令サイクルでようやく1回の演算にとどまる。これが性能向上の足かせとなる。

【0008】[課題2] プログラムサイズの増大

演算処理に必要な命令数が多いため、プログラムサイズ全体が大きくなる。

【0009】[課題3] 命令コードの長大化

演算処理を新規に追加するとき、命令コード中に処理内容を示す演算コードを追加して割り当てる必要があるため、処理機能の拡充が命令コード長の長大化に直結する。この結果、課題2が顕著になり、プロセッサのハードウェア規模も大きくなる。

【0010】[課題4] 命令のコンパクト設計の困難性
基本演算以外の演算処理も将来的に追加、拡張されうるため、予め仕様に組み込む必要がある。このため、多くの演算コードの割当てによって命令コード長のコンパクトな設計が困難となる。

【0011】[課題5] 拡張性を考慮したプロセッサ設計の困難性

長期的な演算処理の追加・拡張をすべて考慮した設計自体極めて困難であり、追加等は自ずと限定される。柔軟かつ拡張性を保持したプロセッサの設計は非常に困難である。

【0012】これらの課題を一部解決する技術として、日本電気(株)のDSPであるμPD77240(商標)を挙げることができる。「ユーザズ・マニュアル μPD77240」(1991年9月版)の66ページ

算)、AND(論理和)、OR(論理積)などの基本的な演算命令に加え、MUL(乗算)、DIV(除算)、MAC(積和演算)、FADD(浮動小数点加算)、FSUB(浮動小数点減算)などの拡張された演算命令を設け、これらの演算命令をプログラム中に記述して実行させるのである。

【0005】しかしこうしたプロセッサでは、演算に必要なデータを演算用データ記憶箇所(例えばレジスタ)に準備する命令と、実際に演算を起動・実行する命令の双方を実行しなければ、所定の演算処理を遂行することができなかった。例えばRISCプロセッサによってメモリに格納されたオペランドデータに対する繰り返し乗算を行うとき、下記のパログラムが考えられるが、ここでは2つのレジスタ(R2、R3)にデータを転送する2つのロード命令および乗算命令が必須であり、1回の乗算に最低3命令の実行が必要であった。

【0006】

```
(R2 ← mem(R0))
(R3 ← mem(R1))
(R2 ← R2 × R3)
```

ジの記載によれば、このDSPの特徴は以下の通りである。

【0013】(1) 通常のALUの他に浮動小数点乗算専用の回路(FMPY)を持つ。前者は明示的な演算命令(ADD、SUBなど)によって演算を起動し、後者はデータ転送命令によって、転送されたデータに対する演算を自動的に起動する。

【0014】(2) FMPYでは、乗算入力用レジスタK、Lの2つのレジスタに転送されるデータが命令サイクルごとに掛け合わされ、乗算結果が1命令サイクル後にFMPYの出力バスに出力され、2命令サイクル後に乗算出力用レジスタMに書き込まれる。

【0015】この結果、データ転送命令のみの記述によって乗算起動が可能となり、上記課題1及び2にある程度の解決を与えるものである。

【0016】

【発明が解決しようとする課題】しかしこのDSPでも、以下の問題が残る。

【0017】1. 課題1、2について

図2はこのDSPの特定命令のフィールド構成図である。この構成は上記マニュアルの14ページに掲載される「オペレーション命令」と総称される命令のフィールド構成で、その概要は以下通りである。

【0018】1. 命令コード長は32ビットである

2. OPは演算処理の種類を示す

3. CNTはDSPの内部状態の変更指示、例えばアドレスレジスタのインクリメント等を行う

4. Q、Pはそれぞれ第1、第2オペランドを表す

5. SRC、DSTはそれぞれ転送命令の転送元、転送先を示す

10

30

40

50

すなわち、原則として32ビット固定の命令の中に、演算、転送、内部状態指示という、性質の異なる命令（以下「個別命令」という）が3つ並記される（同図下を参照）。これは、DSPの多くがマイクロプログラム制御計算機のアーキテクチャを踏襲するためであり、結果的に、ある程度命令コード長が長くなることと引き換えに、合計ステップ数を削減する効果を持つ。この際、各個別命令の記述に必要なビット数の多寡をうまく組み合わせることによって、これらを32ビットに収めるため、このアーキテクチャは課題1、2を解決する方向にある。

【0019】しかし、現実のプログラミングを考えると、この技術による改善効果はさして大きくない。すなわち、効率の最大化を図る場合、常に3つの並記可能な個別命令をセットにする必要がある一方、実際にはそうした配列が不可能だからである。例えば、転送命令のみを数回続けなければならない場合、実質的に上記SRC、DSTのフィールドのみ（計10ビット）を記述するために、毎回32ビットの命令コードが必要となり、この間メモリの使用効率は約1/3となる。この結果、再度課題1、2が問題として現れる。

【0020】2. 課題3～5について

このDSPでは、FMPYによる演算が乗算に限られている。しかし将来別の演算が追加される場合、なんらかの方法で、複数の演算から所期の演算を選択する必要がある。図2の構成を採る限り、こうした選択はOPに新たな個別命令を追加することによって行われるが、こうした対処は、データ転送に関する個別命令のみによって乗算が実行されるというFMPYの設計思想に沿わないだけでなく、課題3～5に直結するおそれがある。もともと、このDSPはこれらの課題を念頭において設計されたものではない点に注意すべきである。

【0021】以上が課題1～5とDSPの関係である。なお、本発明の主用途であるRICSプロセッサを考えた場合、このDSPにはさらに次の課題が考えられる。

【0022】【課題6】プログラミングの困難性

FMPYによれば、常時K、Lレジスタに対する乗算が行われており、2サイクル前の計算結果がMレジスタに現れる。計算結果は参照の有無に関係なく1サイクル毎に更新されるため、アセンブリ言語を使用するプログラムは、Mレジスタから所望の計算結果を読み出す際、タイミングに細心の注意を払わねばならない。一方、高級言語を使用する場合は、プログラムの意図した読み出しタイミングを実現するコンパイラの開発が極めて困難である。

【0023】また、仮にタイミングの問題を解決したとしても、計算結果の読み出し前に割込みが発生すれば別の問題が発生する。割込み処理が終了するまでに、必要な計算結果が失われうるためである。乗算前に割込み禁止を設定する方法もあるが、（1）プログラムが複雑に

なり、（2）割込みの禁止許可はシステムレベルで判断すべき問題であり、（3）FMPYの目的である即時性が失われる、の諸点から現実的でない。

【0024】【課題7】アーキテクチャに起因する制限
このDSPでは、K、L、Mレジスタセットを1つしか持たず、これとFMPYが組として設けられているため、仮に新たな演算（例えば加算）を追加する場合、それに応じて別のレジスタセット（例えばK₂、L₂、M₂）を追加しなければならない。RICSプロセッサでは、画像処理やCGなどの科学技術計算を実施するために、加減乗除、三角関数等各種演算が必要であり、演算ごとに32ビットレジスタのセットを増設していくとすればハードウェアの大規模化を招く。

【0025】【課題8】ハードウェアの規模
このDSPでは、K、Lレジスタに同時にデータを転送することによって高速化を図っている。このため、これらのレジスタの入力口に二重構造の32ビットデータバスを採用する。従ってこの部分のハードウェアが複雑であり、回路規模も大きい。コンパクト設計、製品コストの観点からは改善することが望ましい。

【0026】本発明の目的

本発明の目的は、従来一般的に問題となった課題1～5のみならず、上記したDSPの課題6～8をも解決するデータ処理装置を提供することにある。

【0027】

【課題を解決するための手段】本発明のデータ処理装置は、演算命令の実行によって所定の演算処理を起動する第1演算手段と、所定の入力用レジスタ及び出力用レジスタを含み、データ転送命令によるアクセスが可能な複数のレジスタと、データ転送命令の実行によって前記入力用レジスタに対するデータ転送が行われたことを契機としてそのデータに対する所定の演算処理を起動し、その演算結果データを前記出力用レジスタに書き込む第2演算手段と、前記演算結果データの読み出しが行われるまで、前記出力用レジスタに対する後続の演算結果データの書き込みを待たせる書込停止手段とを備える。

【0028】本発明は、命令の実行が前記演算結果データの読み出しを伴うとき、所望の演算結果データが前記出力用レジスタに書き込まれるまで読み出しを待たせる読出停止手段を含む。

【0029】本発明は、前記書込停止手段によって書き込みが待たされているときに、前記入力用レジスタに対するデータ転送が行われた場合、前記第2演算手段による演算処理の起動を待たせる演算停止手段を含む。

【0030】本発明は、前記入力用レジスタに有効なデータが保持されているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送を待たせる転送停止手段を含む。

【0031】本発明は、前記入力用レジスタ及び出力用レジスタの組合せによる複数のレジスタセットを備え、

さらにこれら複数のレジスタセットから演算処理を行うべきレジスタセットを指定する指定手段を有する。

【0032】前記第2演算手段は、それぞれが異なる演算を行う複数の演算器を有し、本発明の装置は、これら複数の演算器のうちデータ転送が行われたとき実際に演算処理を起動すべきものを指定する指定手段を有する。

【0033】前記指定手段は、前記入力用レジスタおよび出力用レジスタとは別に設けられた制御レジスタである。

【0034】前記第2演算手段は、それぞれが異なる演算を行う複数の演算器を有し、本発明の装置は、前記複数の演算器のうち1つの演算器の演算結果データを別の演算器へ入力する手段と、前記出力用レジスタに書き込まれたデータを前記別の演算器に入力する手段と、前記別の演算器の演算結果データを前記出力用レジスタに再帰的に入力する手段とを有する。

【0035】本発明は、前記入力用レジスタを複数備え、これらの入力用レジスタに対して順次データ転送がなされるとき、1回の演算処理に必要な複数のデータが入力用レジスタに揃うタイミングを検出する検出手段と、これらのデータが揃うまで、前記第2演算手段による演算処理の起動を待たせる演算停止手段を含む。

【0036】本発明は、前記入力用レジスタと前記第2演算手段との間に介挿され、前記入力用レジスタに転送されたデータを順次記憶するFIFOメモリと、前記FIFOメモリからのデータ読み出しを制御する制御手段とを有する。

【0037】

【作用】本発明のデータ処理装置によれば、第1演算手段において演算命令の実行により、所定の演算処理が起動される。一方、データ転送命令の実行によって入力用レジスタに対するデータ転送が行われたとき、第2演算手段においてそのデータに対する所定の演算処理が起動され、その演算結果データが出力用レジスタに書き込まれる。この際、演算結果データの読み出しが行われるまで、出力用レジスタに対する後続の演算結果データの書き込みが待たされる。

【0038】別の態様では、命令の実行が前記演算結果データの読み出しを伴うとき、所望の演算結果データが前記出力用レジスタに書き込まれるまで読み出しが待たされる。

【0039】また別の態様では、前記書き込みが待たされているときに、前記入力用レジスタに対するデータ転送が行われた場合、前記第2演算手段による演算処理の起動が待たされる。

【0040】さらに別の態様では、前記入力用レジスタに有効なデータが保持されているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送が待たされる。

【0041】前記入力用レジスタ及び出力用レジスタの

組合せによるレジスタセットが複数存在する場合は、これら複数のレジスタセットから演算処理を行うべきレジスタセットが指定される。

【0042】前記第2演算手段がそれぞれ異なる演算を行う複数の演算器を有する場合は、これら複数の演算器のうちデータ転送が行われたとき実際に演算処理を起動すべきものが指定される。

【0043】この指定は、前記入力用レジスタおよび出力用レジスタとは別に設けられた制御レジスタによって行われる。

【0044】この場合、別の態様によれば、前記複数の演算器のうち1つの演算器の演算結果データが別の演算器へ入力され、前記出力用レジスタに書き込まれたデータが前記別の演算器に入力され、前記別の演算器の演算結果データが前記出力用レジスタに再帰的に入力される。

【0045】前記入力用レジスタが複数存在する場合、これらの入力用レジスタに対して順次データ転送がなされるとき、1回の演算処理に必要な複数のデータが入力用レジスタに揃うまで、前記第2演算手段による演算処理の起動が待たされる。

【0046】前記入力用レジスタと前記第2演算手段との間にFIFOメモリが介挿される場合は、このメモリが前記入力用レジスタに転送されたデータを順次記憶する。記憶されたデータは、必要なタイミングで順次読み出される。

【0047】より具体的には、演算処理に必要な複数のデータが前記FIFOに揃ったとき前記FIFOメモリからデータの読み出しが行われ、前記FIFOメモリからデータが読み出されたとき、前記第2演算手段による演算処理の起動が許可される。

【0048】

【実施例】以下、本発明の好適な実施例を適宜図面を参照しながら説明する。

【0049】実施例1、本発明のデータ処理装置では、図1同様の演算部における演算が演算命令によって起動される一方、新たな演算部（以下、第2演算部という）における演算処理が特定のレジスタに対するデータ転送命令のみによって起動される。

【0050】図3は本実施例に係るデータ処理装置の全体構成図である。

【0051】同図において、メモリ1には命令およびデータが格納される。命令は命令バス1A、データはデータバス1Bを介してアクセスされる。命令フェッチ部2はメモリ1からデータ転送命令や演算命令などの命令を順次フェッチし、信号線2Aを介してこれを命令デコード部3に渡す。

【0052】命令デコード部3では命令の種類に従って命令コードを所定のフィールドに分割し、処理すべき演算の種類を示すオペレーション、命令コード内に埋め込

まれたオペランドである即値オペランド、ソースレジスタ番号、ディスティネーションレジスタ番号などを抽出する。オペレーションは信号線3Bを介して演算部5へ送られ、レジスタ群4は信号線3Aを介してアクセスされる。

【0053】次にレジスタ群4からソースレジスタ番号で指定されたレジスタの内容が読み出され、信号線4Dを介して演算部5へ送られる。オペレーションが加減算などの演算命令を示していれば、ソースオペランドに対し、演算部5で通常の演算が行われる。演算結果は信号線5Aを介してデータアクセス部6へ送られ、信号線6Aを介してレジスタ群4中のディスティネーションレジスタ番号で示されるレジスタに書き戻される。

【0054】一方、オペレーションがメモリへのストア命令なら、演算部5によって生成されるデータアドレスと書き込みデータにより、データバス1Bを介してメモリ1にデータが書き込まれる。また、ロード命令なら、演算部5によって生成されるデータアドレスに従ってメモリ1の内容が読み出され、レジスタ群4中の指定されたレジスタに書き込まれる。

【0055】以上の構成は従来の装置とほぼ同様であるが、本実施例の特徴はレジスタ群4に接続された第2演算部7にある。

【0056】図4は図3のレジスタ群4と第2演算部7の接続を示す図である。同図において、R0~3およびSR0~2は作業用レジスタであり、ロード命令、ストア命令などのデータ転送命令によってアクセスされる。R0~3は汎用の作業用レジスタ、SR0~2は第2演算部7による使用が可能なデータ格納用レジスタである。SR0、1、2はそれぞれデータ線4A、4B、7Aによって第2演算部7と接続される。さらにSR2から信号線4Cにより、後述の出力データ存在フラグの状態が第2演算部7へ送られる。本実施例では、SR0、1を第2演算部7への入力データ格納用、SR2を第2演算部7からの出力データ格納用に用いる。

【0057】図5は第2演算部7の内部構成図である。同図に示す通り、第2演算部7は実際の演算を行う演算回路71と、これを制御する制御回路72からなる。演算回路71は2入力1出力であり、例えば乗算、除算などを行う。同図中、入力データ存在フラグ0、1はそれぞれ処理の対象となるデータがSR0、1に存在するかどうかを示し、存在するときにセット状態、しないときにリセット状態になる。これらフラグの状態は信号線4Af、4Bfを介して制御回路72に与えられる。

【0058】SR0、1から読み出されたデータ自体は演算回路71に与えられる。演算回路71に有効な演算結果が生じたとき、制御回路72は演算結果存在フラグをセットし、信号線7Aeに出力する。演算回路71は演算結果データをデータ線7Adに出力する。信号線7Aeとデータ線7Adによってデータ線7Aが構成さ

れ、これらがSR2へ与えられる。なお、制御回路72は制御線72Aにより、演算回路71に対して演算の起動を許可する。

【0059】以下、図3~5を用いて本実施例に特徴的なデータ処理動作を説明する。

【0060】1. SR0へのデータロード

まず「メモリからレジスタSR0へのロード」命令がフェッチされたとする。このときメモリから必要なデータが読み出され、データ線6Aを経てSR0にロードされ、前述の入力データ存在フラグ0がセットされる。データは演算回路71に与えられる。

【0061】2. SR1へのデータロード

つづいて、「メモリからレジスタSR1へのロード」命令がフェッチされると、同様にメモリからデータが読み出され、SR1にロードされる。ここで入力データ存在フラグ1もセットされ、データは演算回路71へ与えられる。

【0062】3. 演算の起動

制御回路72は、入力データ存在フラグ0、1がともにセット状態であることを確認し、制御線72Aを介して演算起動指令を演算回路71に与える。演算回路71は指令に応じて演算を起動、実行する。

【0063】4. SR2に対する演算結果の格納

制御回路72は演算回路71における演算実行に必要な時間の経過後、演算結果存在フラグをセットする。演算結果データとこのフラグはデータ線7Aを介してレジスタSR2に与えられる。

【0064】SR2は、出力データ存在フラグ、すなわち自レジスタに有効な演算結果データが存在するかどうかを示すフラグを持つ。このフラグは初期状態でリセットされており、かつ、リセット状態にある場合に限り演算結果の書き込みが許可される。書き込み後、このフラグはセットされる。出力データ存在フラグがセットされたとき、演算結果存在フラグはリセットされる。

【0065】5. SR2からメモリへのデータストア
この後、「SR2からメモリへのストア命令」がフェッチされると、データ線4Dを介してSR2の読み出しが行われ、出力データ存在フラグは再度リセットされる。

【0066】以上が一連の処理の例である。以降、制御回路72は同様に、入力データ存在フラグ0、1がともにセット状態にあることを検出するたびに、演算回路71に演算起動指令を送り、演算が完了すれば演算結果存在フラグをセットする。ただし、演算結果存在フラグがセットされたとき、出力データ存在フラグがいまだセット状態にあれば、SR2への演算結果の書き込みを停止する。

【0067】さらに、演算結果存在フラグがセット状態にあり、かつSR2への書き込みも停止されていれば、この間制御回路72は演算回路71に対する演算起動指令の発行を停止する。

【0068】これらの配慮の結果、プログラマまたはコンパイラは所期データの読み出しタイミングに注意を払う必要がなくなる。すなわち本実施例によれば、必要なデータが必要な箇所に準備されたことを確認して、はじめて後続の処理が行われるため、いかなるタイミングで演算結果の読み出し命令を実行しても、いかなるタイミングで入力データをセットする命令を実行しても、正しい処理を行うことができる。この結果、従来の課題6を解決するものである。

【0069】本実施例では、演算部5における演算が従

LD SR0, (R0) (SR0 ← men(R0))

LD SR1, (R1) (SR1 ← men(R1)) 乗算起動

また、別の処理として、メモリに格納された一次元配列データ（すなわちデータストリーム）どうしの積を計算し、その結果の一次元配列をメモリ格納する場合を考える。こうした処理は、音声処理や画像処理の応用分野で

LD SR0, (R0++)

LD SR1, (R1)

ADD R1, 8 (R1 ← R1 + 8)

ST SR2, (R2++)

ここで3行目の数値「8」は、ポインタR1に与えるべきシフト量である。このプログラムの場合、2行目のロード命令の実行によって第2演算部7の乗算が起動され、3行目の加算命令の実行によって演算部5の加算が起動され、加算と乗算が同時並列的に処理される。この後、ストア命令により、SR2に書き込まれた乗算結果がメモリにストアされる。このプログラムを従来型のRISCプロセッサで実行する場合、乗算命令（MUL）が必要となり、処理速度とプログラムサイズの面で不利である。

【0072】以上、これらのプログラムから明らかなように、本実施例によれば従来の課題6のみならず、課題1、2をも解決することができる。

【0073】本実施例では、第2演算部7が2入力であり、2つのデータがSR0、1に書き込まれたことを検出して演算を起動したが、例えばSR0にデータが書き込まれた時には入力データ存在フラグ0を出さず、SR1にデータが書き込まれた時に入力データ存在フラグ0、1を同時に出力する構成としてもよい。なお、この演算の場合、例えば定数データをSR0に格納しておけば、SR1へのロード命令のみを連続実行することにより、演算を繰り返し行うことができる。これらの構成によれば、従来の課題8をも同時に解決することができる。

【0074】実施例2、つぎに、実施例1の改良例である実施例2を説明する。実施例1の構成では、例えばSR0、1へのロード命令が連続的に実行されたとき、SR2の読み出しがボトルネックとなり、演算処理全体が停止する場合がある。本実施例はこうした性能低下を回避するものである。

来通り演算命令によって起動される一方、第2演算部7における演算処理はSR0、1へのデータ転送命令のみによって起動される。演算回路71が乗算器のとき、メモリに格納された各々異なるオペランドデータに対する乗算を繰り返し実行する場合、例えば下記のプログラムとなる。ここでは1回の乗算あたり2命令しか必要とせず、処理の高速化とプログラムサイズの縮小が実現される。

【0070】

頻繁に実行されるものである。本実施例では、乗算に第2演算部7、配列アドレスの計算に演算部5を使う。プログラム例を示す。

【0071】

(SR0 ← men(R0), R0 ← R0+1)

(SR1 ← men(R1)) 乗算起動

加算と乗算の並列処理

(mem(R2) ← SR2, R2 ← R2+1)

【0075】図6は実施例2の第2演算部7の内部構成図、図7は図6の制御回路72の内部構成図である。これら以外の部分は実施例1と同等でよく、ここでは相違点のみを説明する。

【0076】本実施例の特徴は、演算回路71の2つの入力部に、FIFO（先入れ先出し）メモリ73、74を設け、それぞれSR0、1への連続的書き込みを可能にする点にある。図6に示すごとく、FIFOメモリ73、74は制御回路71から書き込み信号721、722、および読み出し信号723によって制御される。一方、図7に示す通り、制御回路72は以下の構成を持つ。

【0077】(1) 計数回路A724、B727

それぞれFIFOメモリ73、74内に記憶されているデータ数を計数する。これらの回路の初期出力値は0で、それぞれ入力データ存在フラグ0、1がセットされるたびに信号線4Af、4Bfによってカウントアップされ、カウントダウン信号731によってカウントダウンされる。

【0078】(2) 入力制御回路A725

入力データ存在フラグ0がセットされたとき、計数回路A724の示すデータ量がFIFOメモリ73の最大容量未満であれば、そのメモリに対する書き込み信号721を出力し、SR0から送られたデータを書き込む。また、計数回路A724の示すデータ量が0でない限り、常に読み出し要求信号726を出力する。

【0079】(3) 入力制御回路B728

入力制御回路A725同様の制御をFIFOメモリ74について行う。すなわち、入力データ存在フラグ1がセットされたとき、計数回路B727の示すデータ量がF

I F Oメモリ74の最大容量未満なら書き込み信号722を出力し、SR1から送られたデータを書き込む。また、計数回路B727の示すデータ量が0でない限り、常に読み出し要求信号729を出力する。

【0080】(4) 演算制御回路730

読み出し要求信号726、729の両方が出力されたとき、演算対象である2つのデータがレジスタに揃ったと判断する。データが揃い、かつ図6の演算結果存在フラグがリセット状態にあるとき、読み出し信号723をF I F Oメモリ73、74に対して同時に出力し、これらのメモリから2つのデータを読み出す。読み出されたデータは演算回路71に与えられる。このとき同時に、制御線72Aを介して演算回路71に演算の起動を許可する。さらに、計数回路A724、B727に向けてカウントダウン信号731を出力する。

【0081】以上が実施例2の概要である。この構成によれば、データの入力を連続して行うことができる。従って、実施例1の構成による性能の低下を回避することができ、従来の課題6を解決しつつ課題1に対して別の方法による解決を与えるものである。またこの際、SR0、1に対してデータを同時に入力する必要もなく、課題8も解決される。例えば、実施例1で述べた一次元配列どうしの積和計算を考える場合、2つのデータストリームを別々のレジスタに向けて非同期的に投入し、所定のレジスタから非同期的に読み出すようなプログラミングが許され、かつその処理時間も短い。こうした特徴から、本実施例は複雑な科学計算などで大きな効果を発揮する。

【0082】実施例3. つづいて、実施例1の別の改良例である実施例3を説明する。

【0083】図8は本実施例に係るデータ処理装置の全体構成図、図9は図8のレジスタ群4と第2演算部7の接続を示す図である。これらの図はそれぞれ、実施例1の図3、4に対応する。以下、実施例1と同じ構成と動作については説明を省略する。

【0084】実施例3で新設される構成は、

- (1) 出力データ存在フラグの状態をデータアクセス部6に伝える信号線4C'
- (2) 入力データ存在フラグ0、1の状態を命令デコード部3へ伝える信号線4J (信号線4Jは信号線4Afと4Bfの総括名称)
- (3) 本装置によるパイプライン処理動作に対する停止要求信号3Cおよび6B
- (4) 停止要求信号3C、6Bのワイヤード・オアをとる、例えばオープンドレインタイプのゲートg1、g2
- (5) 前記ワイヤード・オアの結果生じる停止信号100

である。この構成による動作の特徴は次の通りである。

【0085】1. SR2からの読み出し命令実行の際、出力データ存在フラグがセットされていなければ、デー

タアクセス部6が停止要求信号6Bを出力し、ゲートg2を経て停止信号100が出力される。この結果、装置は実行待ち状態になる。

【0086】2. SR0にデータを書き込む命令を実行する際に入力データ存在フラグ0がセット状態にあるとき、またはSR1にデータを書き込む命令を実行する際に入力データ存在フラグ1がセット状態にあるとき、命令デコード部3が停止要求信号3Cを出力し、ゲートg1を経て停止信号100が出力される。この結果、装置は実行待ち状態になる。

【0087】以上が実施例3の概要である。本実施例によれば、必要に応じて装置のパイプライン処理自体が停止されるため、演算結果の読み出しタイミングに従来のような制約がない。この結果、前述の課題6を根本的に解決することができる。なお、本実施例においても実施例2同様の配慮を行うことにより、性能の低下を回避することが望ましい。

【0088】実施例4. つづいて実施例4を説明する。図10は本実施例のレジスタ群4と第2演算部7の接続を示す図である。その他の構成はこれまでの実施例と同等である。

【0089】実施例4では、第2演算部7のために使用できる作業用レジスタとして、SR0~5の6本のレジスタを備え、レジスタセットSR0~2またはSR3~5のいずれか一方が選択的に使用されるものとする。レジスタセットの選択は、新たな構成である選択回路41、42とコントロールレジスタ(CTLR)による。選択回路41、42の実体は、例えば2入力1出力のセレクタ群である。選択回路41、42による選択は、コントロールレジスタのselビットによる。レジスタセットが3以上存在するときは、複数のselビット(selフィールド)によって選択を行う。

【0090】この構成より、予めコントロールレジスタによって選択すべきレジスタセットを設定すると、コントロールレジスタから制御信号4Eが選択回路41、42に与えられる。以降、選択されたレジスタセットにより、実施例1同様の処理がなされる。

【0091】以上、本実施例によれば、例えば割込み処理等の実行に通常とは別のレジスタセットを使用することができ、割込み処理中に所望の演算結果が失われるおそれがない。従って、従来の課題7を解決することができる。

【0092】実施例5. 図11は実施例5のレジスタ群4と第2演算部7の接続を示す図である。その他の構成は実施例1等と同等である。

【0093】実施例5では、第2演算部7の中に乗算器700、加算器701の2つの演算器を含み、使用するべき演算器を選択可能としている。このため、選択回路43、44とコントロールレジスタが新設される。すなわち、予めコントロールレジスタのselビットによって

使用すべき演算器を設定することにより、制御信号4Fに従って所期の演算を行うことができる。3以上の演算器が存在するときは複数のselビット（すなわちselフィールド）を設ければよい。

【0094】以上、本実施例によれば、演算器の数が増えてもレジスタの数を増加させる必要がない。また、演算の選択をコントロールレジスタで行うため、演算の指定に例えば図2の命令コード中のOPを使用する必要はなく、命令コード長のコンパクト化が可能となる。また、従来ならば命令によって指定されるべき演算をコントロールレジスタで指定するため、将来の拡張性を確保する意味でも好都合である。これらの結果、従来の課題3～5を解決することができる。

【0095】実施例6、図12は実施例6のレジスタ群4と第2演算部7の接続を示す図である。実施例6では、第2演算部7の中に乗算器702、加算器703の2つの演算器を含み、いずれかの演算器を単独で使用する場合、さらには乗算器と加算器を連結して使用する場合を選択できるよう、選択回路46～48と制御回路45を備える。予め制御回路45に、これら3通りの演算のうち所望の演算を設定すると、制御信号4Gによって選択回路46～48が制御される。

【0096】1. 乗算のみが選択されたとき
SR0、1へのデータ転送命令の実行に応じて乗算が実行され、選択回路48で乗算器702の出力が選択され、SR2に乗算結果が書き込まれる。選択回路46、47の状態は任意である。

```
MOV SR2, 0
L1: LD SR0, (R0++)
    LD SR1, (R1++)
    CB R3, L1
```

従来のRISCプロセッサで同様の処理を実行する場合、加算（ADD）、乗算（MUL）などの演算命令が不可欠であるが、この例では不要である。従って、本実施例でも従来の課題1、2を解決することができる。

【0101】以上、各実施例で本発明のデータ処理装置を説明した。ここでは、第2演算部7が備える演算器として、乗算器、加算器など2入力のを挙げたが、これを例えば逆数演算、三角関数、対数関数などの1入力の演算器に置き換え、1つの入力用レジスタへの転送命令に応じて演算を起動するよう構成してもよい。

【0102】また各実施例では、汎用の作業用レジスタ（R0～3）と、第2演算部7による使用も可能なレジスタ（SR0～5）を同一の機能ブロック内にあるものとして図示したが、例えば本発明をLSI化する際、R0～3を演算部5と同じ回路ブロックに、SR0～5を第2演算部7と同じ回路ブロックに配置してもよい。

【0103】

【発明の効果】本発明のデータ処理装置によれば、データ転送命令の実行によって第2演算手段における所定の

【0097】2. 加算のみが選択されたとき

SR0、1へのデータ転送命令の実行に応じ、選択回路46でSR1側が選択され、選択回路47でSR0側が選択され、加算が実行される。選択回路48では加算器703の出力が選択され、SR2に加算結果が書き込まれる。

【0098】3. 積和演算が選択されたとき

予め、SR2にゼロをロードしてクリアし、選択回路46でSR2側、選択回路47で乗算器702の出力側を選択しておく。ここでSR0、1へデータ転送命令の実行すると、まず乗算器702による乗算が実行され、その結果がSR2に書き込まれる。この乗算結果はSR2から選択回路46を経て加算器703に入力され、一方、乗算結果は選択回路47を経て加算器703へ入力される。この結果、SR0、1の乗算結果とSR2の間で加算が行われる。この後、選択回路48で加算器703の出力が選択され、SR2に加算結果が書き込まれる。

【0099】こうした制御を繰り返し行うことにより、SR0、1へのロード命令を連続的に実行するだけで積和演算処理を行うことができる。以下、このプログラム例を示す。この例は、メモリ上にそれぞれR0、1をポインタとして格納されている2つの1次元配列の各要素の積をとり、それらの総和をSR2に得る処理をループ処理によって記述したものである。

【0100】

```
(SR2 ← 0)
(SR0 ← men(R0), R0 ← R0+1)
(SR1 ← men(R1), R1 ← R1+1)
(R3がゼロでない時L1に分岐)
```

演算処理が起動されるため、少ない命令サイクルでデータ処理を行うことができ、性能向上とプログラムサイズの縮小が可能となる。また、演算結果データの読み出しまで後続の演算結果データの書き込みが待たされるため、プログラミングまたはコンパイラ設計が容易となる。さらに本発明では、演算機能が追加、拡張されても、それが通常のデータ転送命令で起動されるため、別途演算命令を設ける必要がない。従って、命令コードサイズの長大化、ハードウェア規模の増大を抑制することができる。

【0104】別の態様では、命令の実行が演算結果データの読み出しを伴うとき、所望の演算結果データが出力用レジスタに書き込まれるまで読み出しが待たされるため、やはりプログラミング等が容易となる。

【0105】さらにこのとき、前記書き込みが待たされている間に入力用レジスタに対するデータ転送が行われた場合、第2演算手段による演算処理の起動が待たされるため、同様の目的が達成される。

【0106】入力用レジスタに有効なデータが保持され

ているとき、後続のデータ転送命令の実行による該レジスタへのデータ転送が待たされる場合も同様である。

【0107】また本発明では、入力用レジスタ及び出力用レジスタのレジスタセットが複数存在するため、用途に応じて別のレジスタセットを使用することができ、例えば割込み処理等を円滑に行うことができる。

【0108】第2演算手段が複数の演算器を有し、演算の選択が制御レジスタによって行われる場合は、選択内容を命令コード中に反映する必要がないため、命令コードサイズの短縮、将来の機能拡張に対する柔軟な対応が可能となる。

【0109】この場合さらに、ある演算器の演算結果データを別の演算器へ入力することによって複数の演算を連続的に実施することができる。この結果、例えば積和計算など連続的に高速実行することができる。

【0110】一方、入力用レジスタが複数存在するときは、1回の演算処理に必要な複数のデータが入力用レジスタに揃うまで演算処理の起動が待たされるため、プログラミングが容易となる。また、データの同時投入のためのハードウェアも不要であり、ハードウェア規模の縮小が可能となる。

【0111】入力用レジスタと第2演算手段との間にFIFOメモリが介挿される場合は、演算結果データの読み出し等が遅れた場合でもデータの投入を待たせる必要がなく、処理の高速化が可能となる。

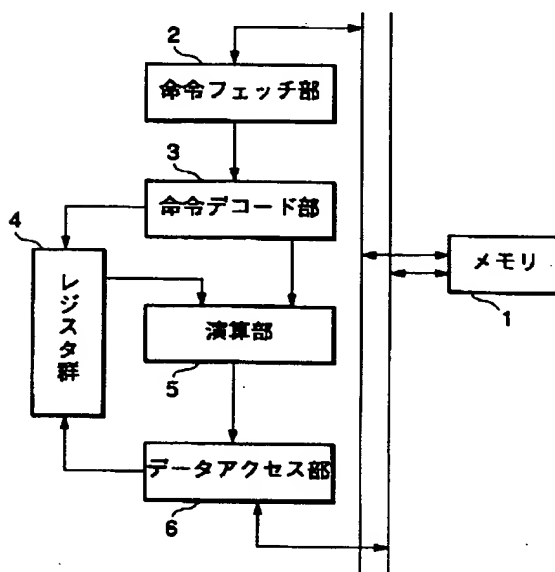
【0112】以上、本発明によれば従来の課題をすべて解決することができる。

【図面の簡単な説明】

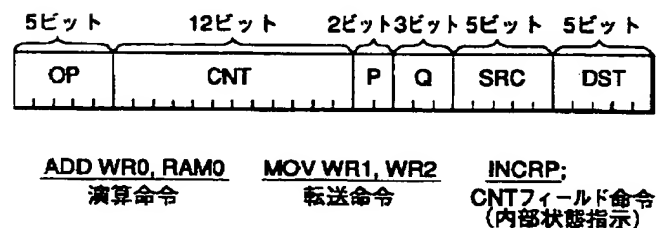
【図1】 RISC方式のプロセッサの構成図である。

【図2】 従来のDSPの特定命令のフィールド構成図

【図1】



【図2】



である。

【図3】 実施例1に係るデータ処理装置の全体構成図である。

【図4】 図3のレジスタ群4と第2演算部7の接続を示す図である。

【図5】 実施例1の第2演算部7の内部構成図である。

【図6】 実施例2の第2演算部7の内部構成図である。

【図7】 図6の制御回路72の内部構成図である。

【図8】 実施例3に係るデータ処理装置の全体構成図である。

【図9】 図8のレジスタ群4と第2演算部7の接続を示す図である。

【図10】 実施例4のレジスタ群4と第2演算部7の接続を示す図である。

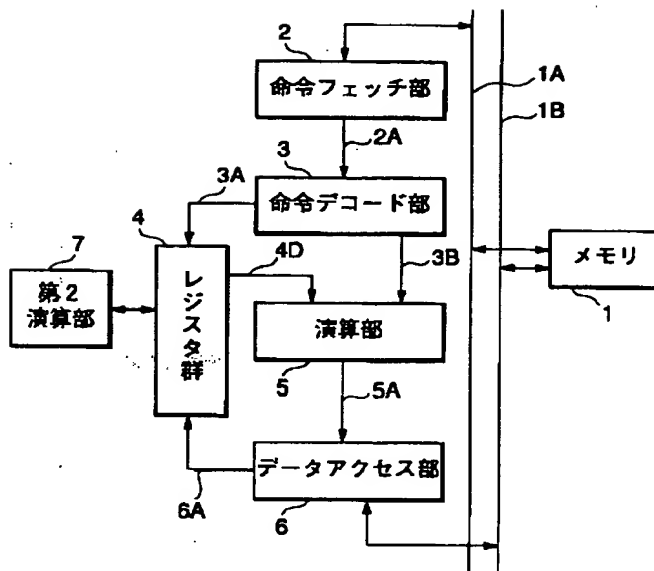
【図11】 実施例5のレジスタ群4と第2演算部7の接続を示す図である。

【図12】 実施例6のレジスタ群4と第2演算部7の接続を示す図である。

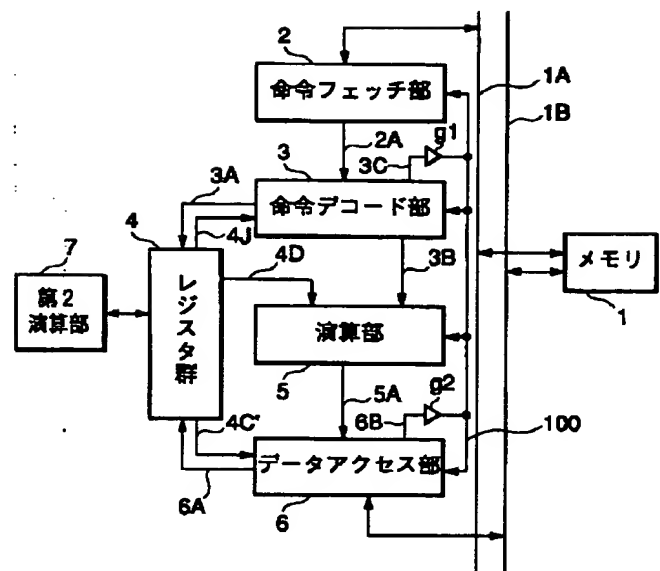
【符号の説明】

1 メモリ、2 命令フェッチ部、3 命令デコード部、4 レジスタ群、5 演算部、6 データアクセス部、7 第2演算部、41~44, 46~48 選択回路、45 制御回路、71 演算回路、72 制御回路、73, 74 FIFO、700, 702 乗算器、701, 703 加算器、724 計数回路A、725 入力制御回路A、727 計数回路B、728 入力制御回路B、730 演算制御回路。

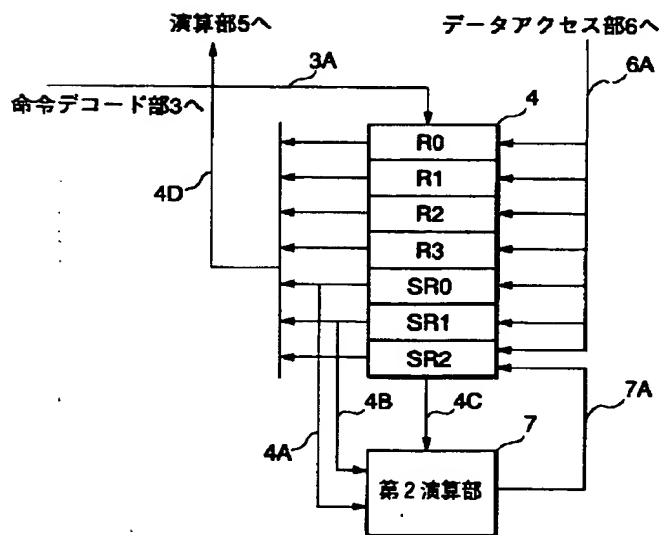
【図 3】



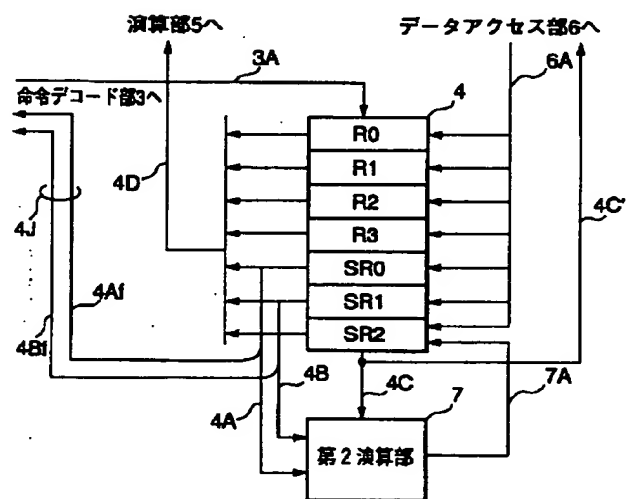
【図 8】



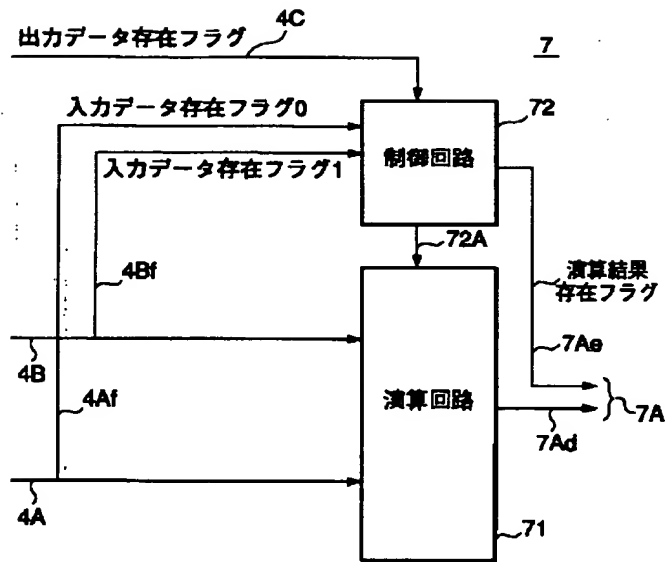
【図 4】



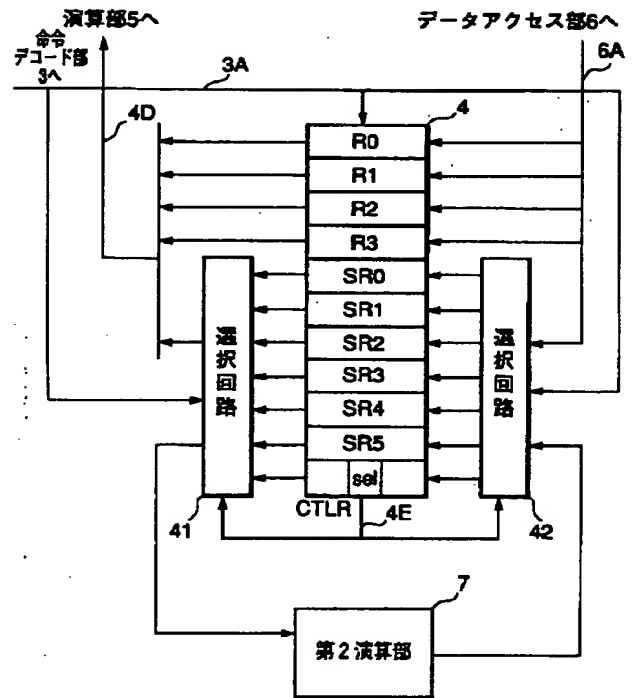
【図 9】



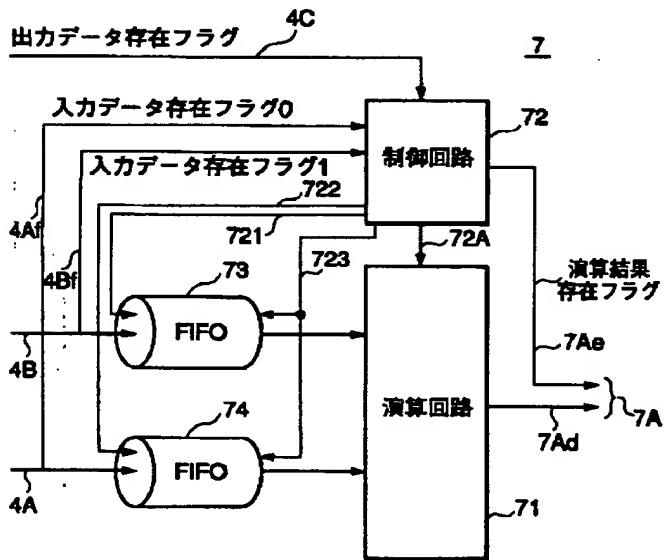
【図5】



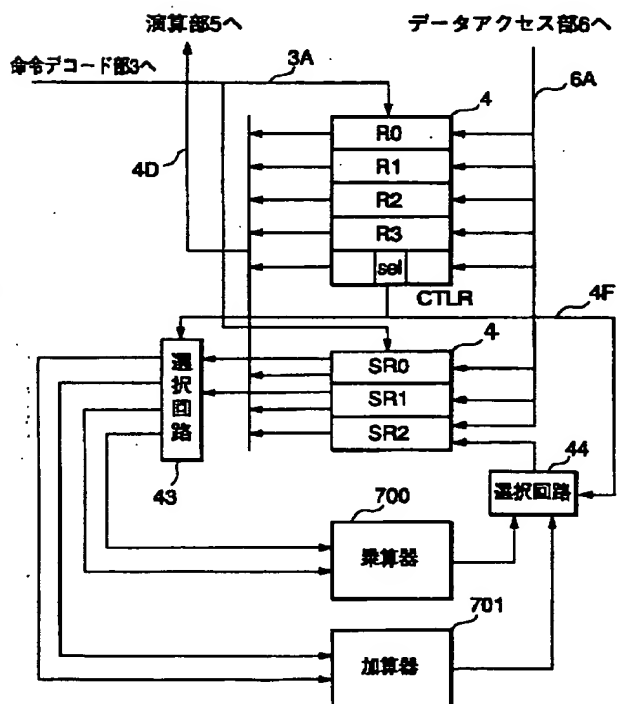
【図10】



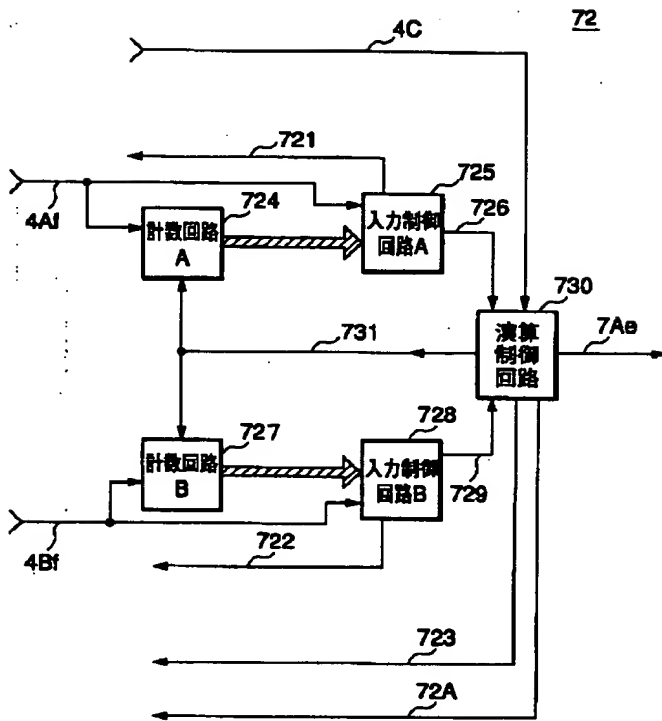
【図6】



【図11】



【図7】



【図12】

